R & D STATUS REPORT

ARPA Order No.:	3771
Contractor:	Caltech
Contract No.:	N00014-79-C-0597 Amount funded - $3,311,894
Effective Date of Contract:	1 March 1979
Expiration Date of Contract:	31 May 1983
Principal Investigators:	Dr. Charles L. Seitz Dr. Carver Mead Dr. Lennart Johnsson
Telephone Number:	213/356-6569, 213/356-6839
Short Title of Work:	Submicron Systems Architecture
Reporting Period:	October 1 1981 through March 31 1982

CONTRACT NO:

N00014-79-C 0597

DESCRIPTION OF PROGRESS

See Attached Report

CHANGE IN KEY PERSONNEL:

None

SUMMARY OF SUBSTANTIVE INFORMATION DERIVED FROM SPECIAL EVENTS

None

PROBLEMS ENCOUNTERED AND/OR ANTICIPATED:

None

ACTION REQUIRED BY GOVERNMENT:

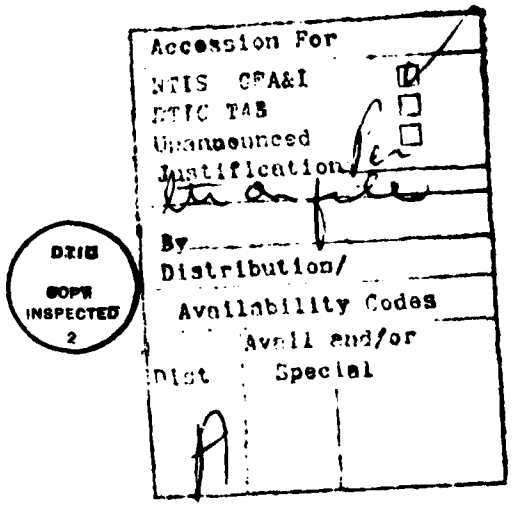
None

FISCAL STATUS:

1. Amount currently provided by contract: $3,311,894

2. Expenditures and commitments as of March 31 1982 2,039,199

3. Funds required to complete work: 1,272,695



CALIFORNIA INSTITUTE OF TECHNOLOGY

Computer Science

Report on

SUBMICRON SYSTEMS ARCHITECTURE PROJECT

May 1982

Co-principal investigators: Charles L. Seitz, Carver A. Mead, Lennart Johnsson

Other faculty: Randal Bryant, Jim Kajiya, Alain Martin, Martin Rem

Ph.D. students: Marina Chen, Young-Il Choo, Erik DeBenedictis, Dick Lang, Bob Lewis, Peggy Li, Sheue-Ling Lien, Mike Ullner, Dan Whelan, Doug Whiting

M.S. students: Bill Athas, Chao-Lin Chiang, Howard Derby, Eric Holstege, Chris Kingsley, Chris Lutz, Parveen Shukla, Craig Steele

The Tree Machine Project

Software:

The basic tree machine software (assembler, loader, simulator) is running well. The simulation demands extensive machine resources. The maximum binary tree that can be simulated on the DEC-20 has depth 7. The node processor being simulated has the following characterisitics:

1. 12-bit Data Path
   a. Three 12-bit I/O ports with flags indicating empty or full of that port
   b. 16 12-bit general purpose registers
   c. A bit-slice ALU with flags (Zero, Carry, Negative, Overflow)
   d. A left-right shifter to do one-bit shifting or rotating
   e. A 12 bit Program Counter(PC) with adder to allow 12-bit address space and/or up to 4K memory
   f. A 12-bit Memory Address Register (MAR) and a 4-bit Memory Data Register (MDR) for memory access, the size of each memory address is 4 bits, i.e. a nibble
   g. two 12-bit Instruction Register (IR) to store the instruction up to 6 nibbles 2

2. Instruction Set
   The instructions are in the unit of nibbles. The length of instructions are varied from 2 to 6 nibbles. The instructions can be grouped into six categories:

   a. memory accessing operations: STORE and LOAD
   b. arithmetical and logical operations: unary and binary operations with registers

c. conditional and unconditional jump: JUMP and BRANCH on true or false
d. subroutine call: CALL
e. input/output operations: IN, OUT and In/Dispatch
f. Status check operations: Set Flags or Store flags

3. Control
The control part is implemented by vertical microgramming. Five PLA's are used. A FETCH PLA fetches an instruction from memory and stores the output into two instruction registers. This PLA then gives control to the EXE PLA. The EXE PLA decodes the opcode, keeps track of the microsteps of each instruction, provides bus addresses and initiates the ALU PLA for arithmetic operations and shift operations, the BRANCH PLA for the branch and jump operations, the I/O PLA for the I/O instructions and the FETCH PLA for the memory access operations. When an instruction is executed, the EXE PLA gives control back to the FETCH PLA that fetches the next instruction.

Documentation: Li, Peggy, "The Tree Machine Operating System", Computer Science, Caltech, internal document 4618:TR:81, July 1981.

Hardware:

An attempt to generate a layout of the processor as defined above using Bristle Blocks for the data path and RELAY for the PLAs and the interconnections have been unsuccessful due to the size of the chip and memory limitations on the DEC-20. Partitioning of the data path and limiting the sizes of the PLAs also turned out to be a difficult path to pursue. New data path and finite state machine compilers are under development.

Meanwhile, the basic project to lay out the processor using Earl and some advanced circuit techniques continues. We made a decision to change the organization of the machine from 12-bit to 16-bit words, and to read the instructions in parallel 16-bit words instead of serially by 4-bit nibbles. The main reason for this change was that we are now confident of our ability to put both the processor and a reasonable amount of storage on one chip with a 4 micron buried-contact run. Previously we felt we must use a separate 1K by 4 standard storage part, and a design similar to what Sally Browning originally proposed with a narrow path to storage fit this scheme well. However, it is not necessary and certainly harmful to performance to use such a narrow data path internal to the chip. A secondary reason for this change is that it simplified and regularized the processor in several ways: The conversions between 4-bit-serial and 12-bit parallel formats accomplished by a separate 4-bit path through the parallel setions goes away, and the length of the microcode and amount of state in the main microcode sequencer is reduced.

The existing assembler and loader are easily adaptable to this design change. The communication remains unchanged except for the word length change.

The logic design is complete, and the microcode written and simulated. Critical circuits have been verified by analysis or SPICE simulation. A variety of dynamic RAM circuits and drivers are being sent to fabrication to verify their operation. The logic design is complete, and the microcode written and simulated.

Our present estimates of chip area are that the processor will be only about 3 million square lambda (about 12 sq mm at lambda equals 2 microns), and on-chip 8K of dynamic RAM about 2 million square lambda.

Other related efforts in design tools and testing are reported below.

Homogeneous Machine Projects

The COPE Machine:

The simulation and analysis of a distributed, on-the-fly garbage collection algorithm have been completed. The results of this study show that the algorithm scales well and is of acceptable complexity. Technical Report 4724 gives a detailed account of the algorithm and its characteristics. The study of interconnection topologies for homogeneous, object oriented machines has been completed. Simulation results and observations will be published in a forthcoming thesis entitled "The Extension of Object-Oriented Languages to a Homogeneous, Concurrent Architecture". This document also describes a concurrent programming style centered around the Simula object concept.

Documentation: Lang, Dick, "A Distributed Class Object Processing Architecture," Computer Science, Caltech, internal document 4199:DF:81, February 1981.

Lang, Dick, "Concurrent, Asynchronous Garbage Collection Among Cooperating Processors", Computer Science, Caltech, Technical Report 4724, February 1982.

Lang, Dick, "The Extension of Object Oriented Languages to an Homogeneous Concurrent Architect," Ph.D. Thesis, Computer Science, Caltech, 1982 - to appear shortly.

The 6-Cube Homogeneous Machine:

Caltech is currently building a concurrent computing system consisting of 64 processing elements organized in a Boolean 6-cube, with each processor communicating to its neighbors through asynchronous, bidirectional buffers. This effort has drawn on some very useful interest and collaboration in the software from a group in High Energy Physics at Caltech. The processing nodes are based on the 8086/8087 with 128K storage, and are thus somewhat oriented toward floating-point numerical computation.

The hardware for the main CPUs has been completely verified, and a 3-node machine with wire-wrap cards is running programs. The node processor is now ready for PC layout.

A C compiler is now functional, although not completely debugged, and fully supports the 8086 and 8087.

The dedicated host that controls the 6-cube is operational, but some additional boards are being designed and constructed.

Benchmarks have been run on the operational node processors and the dedicated host. These benchmarks indicate performance as 1/6th of a VAX with a 5 MHz clock rate. All of the hardware is presently available at 10 MHz except the

8087, and it is expected that 10 MHz is achievable. At 10 MHz the performance per node is 1/3 of a VAX and for the entire machine would be 21 times a VAX. (The expected processor utilization is greater than 80% for some applications.)

Test programs have been written and run. At present, these programs have utilized only one processor. (The purpose of these programs has been to debug the compiler.) It is expected that communicating concurrent programs will begin running within a month or so.

Much attention and effort is being devoted to the communication issues. A communications chip specialized for serial communications in a binary N-cube is being designed.

Programming Notations and Runtime Systems:

There are many programming paradigms applicable to homogeneous machines. Applications such as grid-point computations can be programmed in almost conventional ways, and our collaborators in physics are using this approach. On the opposite extreme of difficulty are schemes based on functional or application notations.

It seems most feasible now based on Lang's work (and others, such as Hewitt) to implement a programming model on homogeneous machines in which a program consists of a number of sequential processes which communicate by explicit message passing. During execution each processor would be responsible for a number (e.g. 100) of processes and would schedule them for execution based on the arrival of messages. This dynamic scheduling relieves the programmer from specifying the ordering of individual events for each processor, such as our physics collaborators must do, and allows more flexibility in the program execution.

Bill Athas is studying a LISP-like system as a general programming environment for the machine. LISP offers certain features such as inherent concurrency and a copy and destroy attitude towards data which may prove it suitable for such machines. Peter Henderson's LISPKIT compiler kit has been implemented in Pascal. Using a Pascal cross-compiler the system will be ported from the DEC-System 20/60 to the Homogeneous Machine. Currently procedures are being defined to evaluate the amount of concurrency achievable from sample LISPKIT programs.

Documentation: Athas Bill, "A Proposal For A Virtual Homogeneous Machine", Computer Science, Caltech, internal document.

DeBenedictis, Erik, "A Communications Operating System for the Homogeneous Machine", Computer Science, Caltech, Technical Memo #4707, January 1982.

DeBenedicitis, Erik, and Charles L. Seitz, "Homogeneous Machine Technical Plan", Computer Science, Caltech, Techical Memo #4705, January 1982.

Computational Arrays

Concurrent versions of a few familiar numerical methods in computational linear algebra have been derived. The algorithms have the same numerical properties as their sequential counterpart. The concurrent algorithms can be

implemented efficiently in one- or two-dimensional pipelined arrays, i.e., every processing element performs a useful computation every cycle or step. The algorithms that have been devised do not necessarily maximize concurrency or minimize the number of cycles for the computation of the solution. Such algorithms often implies complex communication topologies.

The set of concurrent algorithms that has been devised now includes Gaussian elimination with or without partial pivoting, Cholesky's, Crout's, Doolittle's, Given's and Householder's methods for the solution of linear systems of equations. Gaussian elimination with partial pivoting can only use a one-dimensional pipelined array efficiently due to the data dependent data flow. Householder's method has a data independent data flow, but can nevertheless only use a one-dimensional array efficiently due to the ordering of operations required by the algorithm.

If the problem to be solved fits within the array of processing elements only a few registers per processing elements is usually sufficient for data storage. Linear arrays obviously will require storage of one column of the matrix for each processing element. If the problem is too large to allow for a full instantiation in space, storage of data is trivially needed. Storage external to the array implies block algorithms for the total computation task. By partitioning the storage among the processing elements entirely different algorithms are of interest. These issues are now being investigated.

In addition to the algorithms already studied concurrent algorithms are also being devised for eigenvalue problems.

Documentation: Johnsson, Lennart, "VLSI Algorithms for Doolittle's, Crout's and Cholesky's Methods", internal document, Caltech, April 1982.

Johnsson, Lennart, "Pipelined linear Equation Solvers and VLSI", Microelectronics 1982, May 12-14, 1982.

Johnsson, Lennart, "A Computational Array for the QR-method", The MIT Conference on Advanced Research in VLSI, January 25-27, 1982.

Johnsson, Lennart, "Computational Arrays for Band Matrix Equations", internal document 4287:TM:81, Computer Science, Caltech, May 1981.

Johnsson, Lennart and Danny Cohen, "Computational Arrays for the Discrete Fourier Transform," Proceedings of the Twenty-Second Computer Society International Conference, COMPCON 81, February 1981.

Switch-Level Model for MOS Logic Design

The simulator MOSSIM II has been implemented in Mainsail and is now being tested on designs at Caltech and several other sites. This program improves on previous simulators such as MOSSIM and TSIM (Terman) in its generality, accuracy, and speed. A large variety of ratioed, complementary, and charge-sharing circuits can be simulated in a very straightforward way. MOSSIM II also simulates the effect of the X state (representing an invalid or ambiguous logic value) in a very accurate and consistent way. Algorithms have been implemented which utilize this state to detect a variety of design errors including: races, dynamic charge sharing, unrestored logic levels in CMOS, and

loss of dynamically-stored charge. All of these tests are applied by setting mode switches in the simulator and therefore can utilize the same network description and test sequences as the normal logic simulation. A copy of the user's manual is available on request.

Documentation: Bryant, Randal E., "Switch-Level Modeling of MOS Digital Circuits", International Symposium on Circuits and Systems, IEEE, May, 1982.

Bryant, R., Schuster, M., Whiting, D., "MOSSIM II: A Switch-Level Simulator for MOS LSI, User's Manual", unpublished manual, Caltech, 1982.

Bryant, Randal E., "A Switch-Level Model of MOS Logic Circuits", Proceedings of the First International Conference on VLSI, VLSI 81, Academic Press, 1981, pp. 329-340.

Bryant, Randal E., "A Switch-Level Simulation Model for Integrated Logic Circuits", Laboratory for Computer Science, MIT, Technical Report MIT/LCS/TR-259, March 1981.

Testing

Testing efforts previously reported are continuing, and a Ph.D. thesis reporting this work will appear over the summer. Recent developments include:

- Development of two packages of application software for the test system: one based on C and one for the LSI-11.
- A user's manual for the test system is now available.
- The testers have been packaged in a reasonable manner, and are in use by the VLSI Design class.

Documentation: DeBenedictis, Erik, "A Methodology for Describing Test Specifications", Computer Science, Caltech, internal document 4685:DF:81, November 1981

DeBenedictis, Erik, "FIFI Test System, Preliminary User's Manual," Computer Science, Caltech, internal document 4270:TR:81, April 1981.

DeBenedictis, Erik, "A Preliminary Report on the Caltech ARPA tester project," Computer Science, Caltech, Technical Report 4061:TR:81, April 1980.

DeBenedictis, Erik, "Techniques for Testing Integrated Circuits," draft Ph.D. Thesis, to appear shortly.

Local Network and Designer Workstations

A display system architecture which has rectangular area filling as its primitive operation is presented. It has been shown that lines can be drawn significantly faster with this architecture than with a pixel display system. The rendition of filled boxes is also faster showing an $O(n^2)$ speed improvement. Furthermore filled polygons can be rendered with an $O(n)$ speed improvement. A custom VLSI integrated circuit is currently being designed to

implement this rectangular area filling architecture and at the same time reduce the display memory system video refresh bandwidth requirements.

Also in the graphics area a simple Multibus compatible color graphics system has been designed. Our hope is that this system can be used with the SUN Workstation processor and Ethernet cards to form a CAD workstation. Currently a prototype is running.

In the networking area, our VAX is now on the ARPANET. Work is in progress on laying a departmental Ethernet. This net will connect the DEC System 2060 and the VAX 11/780 initially and will employ the IP/TCP protocols. Five EtherTIPs based on SUN systems are being constructed and will provide the department with terminal line capabilities. In the future, other machines (VAX 11/750, 11/34) will be connected and hopefully at sometime in the future CAD workstations will be connected up. Our plans call for us to have the 2060 and the VAX talking by the beginning of the summer.

Documentation: Whelan, Dan, "A Rectangular Area Filling Display System Architecture", to be presented at SIGGRAPH-82, July, 1982. Also to appear in Computer Grahics.

Whelan, Dan and Ray Eskenazi, "An Inexpensive Multibus Color Frame Buffer", Computer Science, Caltech, internal document 4334:TR:81, June, 1981.

Whelan, Dan, "A Versatile Ethernet Interface", Computer Science, Caltech, Technical Report 4654:TR:81, July 1981.

Earl

Earl has now been developed to the point that it is useful for circuit designers. The CS/EE 181/186 VLSI Design class used it this year for the class projects, producing 26 projects that have been fabricated and are now being tested. Due to the class use Earl is now fairly well debugged.

There is a growing body of cells that can be shared by designers. There is a set of pad drivers and receivers that take lambda into account, so that we can take advantage of different processes, even if the pad sizes must remain the same. The microprocessor group has developed a PLA generator, including superbuffer input drivers that take little more room than the standard pair of inverters. More cells will be collected and documented to increase the utility of the system.

Earl has been distributed to numerous universities and companies.

Documentation: Kingsley, Chris, "Earl: An Integrated Circuit Design Language", Computer Science, Caltech, Technical Memo #4710, February 1982.

Self-timed Systems

Chip designs:

New developments in the work previously reported on the self-timed multiprocessor communication chips can be summarized as follows: An improved method has been found for distributing the masked address of each IP chip to

its address comparator and to the F-box RAM. The address and address mask are simply sent through the transmitter queue as a special message, and the address comparator picks these values out of the message. In the F-box, a special type of decoder on the RAM can be used to expand the masked address into the full alias set; namely, each decoder cell does a masked address decode. A group of seven students from the CS/EE 186 VLSI Design class is beginning layout of the IP chip, and we anticipate completion of that project by the end of the academic year.

Documentation: Whiting, Doug, "A Self-Timed Chip Set For Multiprocessor Communication", Computer Science, Caltech, Technical Report 5000:TR:82, February, 1982.

Theory:

A visiting graduate student from Eindhoven, Jo Ebergen, produced a linearized version of the s-net notation for signals that also parallels the test language developed by DeBenedictis. This notation allows a more concise statement of closure demonstrations, for example:

C1 = [a:=1, b:=1; 1=:x; a:=0, b:=0; 1=:x]

C2 = [x:=1, c:=1; 1=:d; x:=0, c:=0; 0=:d]

represents an interconnection of 2 2-input C-elements, both function and domain, and

C3 = [a:=1, b:=1, c:=1; 1=:d; a:=0, b:=0, c:=0; 0=:d]

the function and domain (environment) of a 3-input c-element. The composition (C1 0 C2) does not equal C3, since its domain is broader, but is a valid implementation of C3 since (C1 0 C2) 0 C3* = [ ], where * (dual) represents exchanging system and environment (or function and domain). A new proof of the weak conditions theorem was expressed in this notation.

It is our hope that a genralization of this notation may be useful as a message-passing programming notation as well as for signal-described systems.

Documentation: Seitz, Charles L. "s-nets" -- course notes for CS181b, and op cit

Characterization of Deadlock Free Resource Contention

An important class of deadlock problems is caused by conflict over shared resources. Resource sharing without a global scheduler is considered in this research. The only scheduling allowed is the delay of a process requiring a resource already in use. The acquisition of resources is concurrent and asynchronous, and no global information is available. This type of "minimal scheduling" is particularly important in VLSI, where the requirement to have global scheduler would seriously degrade its computing potential.

A theorem has been proved for necessary and sufficient conditions for collections of synchronized processes to be free from deadlock in such concurrent systems. The general theorem which characterizes the danger of

deadlock in such collections is proved by applying Philip Hall's Theorem on system of distinct representatives (SDR). It is shown that there exists a polynomial time algorithm for testing the deadlock free condition. Also a special case in which all resources are of different types is discussed.

Documentation: Chen, Marina, Rem, Martin, and Graham, Ronald, "A Characterization of Deadlock Free Resource Contentions", Computer Science, Caltech, Technical Report 4684:TR:82, January 1982.

Semantics of Concurrent Systems

A formal representation is developed for concurrent systems ranging from the transistor level up to the level of communicating processes. The research culminates in a single notation and formalism for designing and characterizing these systems.

A single sequential process is represented by a function and data are represented as functions with space and time as their domain. A concurrent program consists of a set of functional equations and its meaning is given by its fixed-point. Once the fixed point is obtained it can be used in the construction of other concurrent programs. Verification techniques for synchronous and asynchronous concurrent systems are presented. The notion of a universal simulator is introduced and implication for general silicon compilation is discussed.

Documentation: Chen, Marina, "A Semantics for Systolic Arrays", Computer Science, Caltech, internal document 4635:DF:81, August 1981.

Chen, Marina, "HARMOS-A Notation for Designing Concurrent Systems", Computer Science, Caltech, internal document, August, 1980.

Concurrency Algebra

Hierarchical Nets:

Analysis of liveness and safeness for general Petri nets is computationally intractable. A synthetic approach is explored where a class of Petri nets, called structured nets, is recursively defined based on the notion of a process - a unit of action with a beginning and an end - to preserve liveness and safeness. Because structured nets are too restrictive, a loosely coupled communication mechanism between processes called mutual exclusion is added. Criterion for liveness and safeness for structured nets with mutual exclusion is presented. A generalization of structured nets with mutual exclusion is recursively defined to form hierarchical nets.

This work will appear shortly as a technical report: Hierarchical Nets.

Net Transformations:

In the study of hierarchical nets the transformations used to define the nets were restricted to preserve liveness and safeness. When this restriction is relaxed we obtain a set of general net transformations that exhibit interesting properties.

The dual of a Petri net is another net where the places and the transitions are interchanged. We define the following pairs of concepts to be dual of each other: place/transition, not safe/not live, merge/division, splitting/refinement. Then it is possible to form eight net transformations by taking a concept from the first set and appending it an operation from the second set: {place, transition} {merge, division, splitting, refinement}.

Let x range over the first set, and y over the second, then we denote a net transformation by x y. For example, if x is 'place' and y is 'merge' then x y is the net transformation 'place merge'. Let x' denote the dual of x. If x y is a net transformation, then x'y is the net transformation obtained by taking the dual of the first component, and similarly for y. Let z denote either not safe or not live, and z' denote the opposite.

Theorem: If x y is z, then x'y is z' and x y' is z' and x'y' is z.

Proof is by constructing the appropriate net transformations and showing that the results hold.

The above result suggests we look more closely at the nature of duality for Petri nets. The major difficulty has been what to do with the marking when one takes the dual of a Petri net. By looking at net transformations partial answers are suggested.

Categorial Formulation of the Shuffle:

In studying the infinite behavior of Petri nets, we need to precisely define the shuffle (ie. the fair merge) of two infinite sequences. The only definition available so far uses maximal and minimal fixed points over some domain. Unsatisfied with this definition which is far from clear as to what is really going on, we are attempting to define it anew using the framework of category theory.

We consider the category STRINGS where the objects are strings which are functions from some ordinal to a set of letters. The morphisms or arrows are string injections which preserve the orderings. Using the notation defined so far we are able to define the notion of a limit of an injective sequence to be the limit of a diagram.

We are now attempting to define the shuffle as the limit of some diagram. Since the shuffle is a set of strings, we need to consider sets of strings as objects of some other category. Once a satisfactory definition is obtained, we may investigate various composition rules in this category. We believe this will give us a very clear picture of the underlying structure of strings in general leading to a theory of strings which can be used as a semantic domain for models of concurrent computation like Petri nets.

Documentation: Choo, Young-il, "Concurrency Algebra: Towards an Algebraic Semantics of Petri Nets," Computer Science, Caltech, internal document 4085:FD:80, December 1980.

PUBLICATIONS, TECHNICAL REPORTS AND INTERNAL MEMORANDAS (ARPA)

Athas Bill, "A Proposal For A Virtual Homogeneous Machine", Computer Science, Caltech, internal document.

Barton Antony F., "A Fault Tolerant Integrated Circuit Memory," Computer Science, Caltech, Technical Report 3761 (Ph.D. Thesis), April 1980.

Browning, Sally A. and Charles L. Seitz, "Communication in a Tree Machine," Proceedings of the Second Caltech Conference on VLSI, January 1981.

Browning, Sally A., "Generating Padding Processors for Arbitrary Fanout Trees", Computer Science, Caltech, internal document #3827, July 1980.

Browning, Sally A., "The Tree Machine: A highly concurrent computing environment," Computer Science, Caltech, Technical Report 3760 (Ph.D. Thesis), January 1980.

Bryant, R. E., "Switch-Level Modeling of MOS Digital Circuits", International Symposium on Circuits and Systems, IEEE, May, 1982.

Bryant, R., Schuster, M., Whiting, D., "MOSSIM II: A Switch-Level Simulator for MOS LSI, User's Manual", unpublished manual, Caltech, 1982.

Bryant, Randal E., "A Switch-Level Model of MOS Logic Circuits", Proceedings of the First International Conference on VLSI, VLSI 81, Academic Press, 1981, pp. 329-340.

Bryant, Randal E., "A Switch-Level Simulation Model for Integrated Logic Circuits", Laboratory for Computer Science, MIT, Technical Report MIT/LCS/TR-259, March 1981.

Carroll, Chris R., "A Smart Memory Array Processor for Two-Layer Path Finding," Proceedings of the Second Caltech Conference on VLSI, January 1981.

Carroll, Chris R., "Hardware Path Finders," Computer Science, Caltech, internal document, September 1980.

Chen, Marina, Rem, Martin, and Graham, Ronald, "A Characterization of Deadlock Free Resource Contentions", Computer Science, Caltech, Technical Report #4684, January 1982.

Chen, Marina, "A Semantics for Systolic Arrays", Computer Science, Caltech, internal document #4635, August 1981.

Chen, Marina, "HARMOS-A Notation for Designing Concurrent Systems", Computer Science, Caltech, internal document #3927, August, 1980. 13

Choo, Young-il, "Concurrency Algebra: Towards an Algebraic Semantics of Petri Nets," Computer Science, Caltech, internal document #4085, December 1980.

DeBenedictis, Erik, "A Communications Operating System for the Homogeneous Machine", Computer Science, Caltech, Technical Memo #4707, January 1982

DeBenedicitis, Erik, "Hogeneous Machine Technical Plan", Computer Science, Caltech, Techical Memo #4705, January 1982.

DeBenedictis, Erik, "A Methodology for Describing Test Specifications", Computer Science, Caltech, internal document #4685, November 1981

DeBenedictis, Erik, "FIFI Test System, Preliminary User's Manual," Computer Science, Caltech, Technical Memo 4270, April 1981.

DeBenedictis, Erik, "A Preliminary Report on the Caltech ARPA tester project," Computer Science, Caltech, Technical Report 4061, April 1980.

DeBenedictis, Erik, "Justification for a Tree Machine", Computer Science, Caltech, internal document #3751, May 1980.

Johnsson, Lennart, "VLSI Algorithms for Doolittle's, Crout's and Cholesky's Methods", internal document, Caltech, April 1982.

Johnsson, Lennart, "Pipelined linear Equation Solvers and VLSI", Microelectronics 1982, May 12-14, 1982.

Johnsson, Lennart, "A Computational Array for the QR-method", The MIT Conference on Advanced Research in VLSI, January 25-27, 1982

Johnsson, Lennart and Danny Cohen, "A Mathematical Approach to Modelling the Flow of Data and Control in Computational Networks", The CMU Conference on VLSI Systems and Computations, Pittsburgh, October 19-21, 1981.

Johnsson, Lennart, "Computational Arrays for Band Matrix Equations", Computer Science, Caltech, internal document #4287, May 1981.

Johnsson, Lennart and Danny Cohen, "Computational Arrays for the Discrete Fourier Transform," Proceedings of the Twenty-Second Computer Society International Conference, COMPCON 81, February 1981.

Johnsson, Lennart, Uri Weiser, Danny Cohen and Alan L. Davis, "Towards a Formal Treatment of VLSI Arrays," Proceedings of the Second Caltech Conference on VLSI, January 1981.

Johnsson, Lennart, "A Note on Householder's Method, Sparse Matrices and Concurrency," Computer Science, Caltech, internal document #4089, December 1980. 14

Johnsson, Lennart, "Gaussian Elimination on Sparse Matrices and Concurrency," Computer Science, Caltech, internal document #4087, December 1980.

Kajiya, J., "Ray Tracing Parametric Patches", SIGGRAPH-82, July 1982.

Kajiya, J. and Mike Ullner, "Filtering High Quality Text for Display on Raster Scan Devices", SIGGRAPH-81, Computer Science, Caltech, April 1981.

Kingsley, Chris, "Earl: An Integrated Circuit Design Language", Computer Science, Caltech, Technical Memo #4710, February 1982.

Lang, Dick, "Concurrent, Asynchronous Garbage Collection Among Cooperating Processors", Computer Science, Caltech, Technical Report 4724, February 1982.

Lang, Dick, "A Distributed Class Object Processing Architecture," Computer Science, Caltech, internal document #4199, February 1981.

Lewis, Robert, "Switching Dynamics", Computer Science, Caltech, Technical Report 4675, October, 1981.

Li, Peggy, "The Tree Machine Operating System", Computer Science, Caltech, internal document 4618, July 1981.

Li, Peggy, "The Logarithm Machine," Computer Science, Caltech, Technical Report 4517 (MS Thesis), Caltech, November 1980.

Lien, Sheue-Ling, "Towards a Theorem Proving Architecture", Computer Science, Caltech, Technical Report 4653, July 1981.

Locanthi, Bart, "The Homogeneous Machine", Computer Science, Caltech, Technical Report 3759 (PhD Thesis), January 1980.

Mead, Carver and Rem, Martin, "Minimum Propagation Delays in VLSI", Proceedings of the Second Caltech Conference on VLSI, January 19-21, 1981, also published as Computer Science Technical Report 4601, 1981.

Rem, Martin and Carver Mead, "A Notation for Designing Restoring Logic Circuitry in CMOS," Proceedings of the Second Caltech Conference on VLSI, January 19-21, 1981, also published as Computer Science Technical Report 4600, 1981.

Rem, Martin, "Communication in a Binary Tree, I and II, Some observations on a tree mapping problem," internal documents, Computer Science, Caltech, June and July 1980.

Rudin, Leonid, " Lambda-Logic", Computer Science, Caltech, Technical Report 4521, May 1981.

Seitz, Charles L, et al, "Proposed instruction set for the tree machine processor," internal document, Computer Science, Caltech, July 1980.

Whelan, Dan, "A Rectangular Area Filling Display System Architecture", to be presented at SIGGRAPH-82, July, 1982. Also to appear in Computer Grahics.

Whelan, Dan, "A Versatile Ethernet Interface", Computer Science, Caltech, Technical Report 4654, July 1981.

Whelan, Dan and Ray Eskenazi, "An Inexpensive Multibus Color Frame Buffer", Computer Science, Caltech, internal document #4334, June, 1981.

Whiting, Doug, "A Self-Timed Chip Set For Multiprocessor Communication", Computer Science, Caltech, Technical Report #5000, February, 1982.

Internal documents may be obtained only from the authors. Technical Reports and Technical Memoranda can be obtained from the Computer Science Librarian, Room 256-80, California Institute of Technology, Pasadena, California 91125. Please identify yourself to the librarian as a member of the ARPA community!